DOCUMENT RESUME

ED 295 638 IR 013 348

AUTHOR Foreman, Kim Hyun-Deok
TITLE Cognitive Style, Cognitive Ability, and the Acquisition of Initial Computer Programming Competence.
PUB DATE Jan 88
NOTE 15p.; In: Proceedings of Selected Research Papers presented at the Annual Meeting of the Association for Educational Communications and Technology (New Orleans, LA, January 14-19, 1988). For the complete proceedings, see IR 013 331.
PUB TYPE Reports - Research/Technical (143) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS *Cognitive Ability; *Cognitive Style; College Students; Computer Literacy; Correlation; *Field Dependence Independence; Higher Education; Logical Thinking; *Programing; Psychological Studies; *Spatial Ability
IDENTIFIERS BASIC Programing Language

ABSTRACT

The subjects for this study, which examined the relationship of cognitive style and ability to the initial acquisition of computer programming competence, were 46 college students enrolled in a computer literacy course. The selected cognitive style (field-independence) and three cognitive abilities (logical reasoning, spatial ability, and direction following) were correlated to five programming component skills: (1) knowledge of BASIC syntax; (2) ability to predict program outcome; (3) ability to design and write programs; (4) ability to debug programs; and (5) ability to modify programs in BASIC. It was found that field-independence, logical reasoning, and direction following were strongly related to most programming skills, while spatial ability was related only to debugging and modification. The text is supplemented by one figure. (26 references) (Author/EW)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED295638

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality

• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy

Cognitive Style, Cognitive Ability, and the Acquisition of Initial Computer Programming Competence

Kim Hyun-Deok Foreman

University of Wisconsin-Madison

Running head: COGNITIVE STYLE AND PROGRAMMING

IR013348

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
Michael R. Simonson

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

Abstract

This study examined the relationship of cognitive style and ability to the initial acquisition of computer programming competence. The selected cognitive style (field-independence) and three cognitive abilities (Logical reasoning, spatial, and direction following) were correlated to five programming component skills. The five programming component skills measured in this study included; (1) knowledge of BASIC syntax, (2) ability to predict program outcomes, (3) ability to design and write programs, (4) ability to debug, and (5) ability to modify programs in BASIC. Forty-six college students in a computer literacy course participated in this study. Results indicated that field-independence, logical reasoning and direction following were strongly relate to most programming skills. Spatial ability was related to only debugging and modification.

Cognitive Style, Cognitive Ability, and the Acquisition of Initial Computer Programming Competence

The curriculum of computer literacy has been controversial since first introduced as a course of study, and remains an unresolved issue. Computer literacy curricula have included one or a combination of following content; computer programming, computer applications, and use of hardware and software. In recent years, the focus of curriculum has shifted from programming language learning to computer application. However, proponents for teaching programming in schools argue that computer technology is a multi-purpose technology. Therefore, the powerful ideas that underlie programming enable students to communicate and interact with a myriad of other computer applications (Soloway, 1987). Most teachers who include computer programming in computer literacy argue that programing provides a hands-on computer experience in an environment where students are in control (Luehrmann, 1981). Programming has become an integral part of many computer literacy curricula.

Programming is complex and novice programmers bring a variety of cognitive skills and styles to the challenge of programming. Individual differences that exist prior to learning new skills are important instructional conditions for planning instructional treatments. However, the relationship between competence in computer programming and cognitive characteristics in learners has not been well researched (Linn, 1985).

Objective

The objective of this paper is to describe the relationship between one cognitive style and five programming component skills, plus, the relationship between three cognitive abilities and five programming component skills. The single cognitive style selected for this study was field independence. The three cognitive ability factors were logical reasoning ability, spatial ability, and direction following. The five components of computer programming success were: syntax, program comprehension, program composition, debugging and program modification.

Background

Considerable research has been conducted to identify factors that are associated with programming outcomes. Findings suggest that there is a positive relationship between programming outcomes and cognitive variables (i.e., general intelligence, mathematical ability, academic achievement). Therefore, it is plausible to hypothesize that individual difference in the initial acquisition of programming competence may be related to cognitive style and cognitive ability.

Cognitive Style

Cognitive style refers to the manner in which individuals process information; how they think (Witkin, Moore, Goodenough, & Cox, 1977). According to cognitive style theory (Ausburn & Ausburn, 1979; Messick, 1984; Kolb, 1984), individuals develop a preferred way of thinking, problem-solving, and interacting with the environment. A commonly identified cognitive style in relation to academic achievement is the field-independent/field -dependent continuum. This style is identified by the Embedded Figures Test (EFT) or Group Embedded Figures Test (GEFT) which measures subject performance on a series of problems in which the subject must find a simple figure in the context of a complex set of figures.

Field-independent learners tend to impose structure on a field if a logical pattern does not exist, whereas field-dependent learners accept the field the way it is (Witkin, Moore, Goodenough, & Cox,1977). In addition, field-independent learners appear to have greater skill in "cognitive restructuring"; that is, in understanding problems and reformulating problems into structured ideas. Skill in cognitive restructuring has been found to be related to success in mathematics and in the physical sciences (Witkin, Moore, Goodenough, & Cox, 1977). Computer programming seems to demand cognitive skills which have been developed by field-independent learners. Programmers often have to reformulate problems into structured ideas and detect irregularities in programming logic. Similarities between cognitive skills used by field-independent learners to solve novel problems, and cognitive requirements for computer programming has led several researchers to hypothesize a relationship. Many have found a moderate but positive relationship between field-independence and computer programming success (Stevens, 1983; Webb, 1985; Bradley, 1986).

Cognitive Ability

Snow (1980) suggests that cognitive ability can be further subdivided into two categories based on cognitive functions. Crystallized ability (general ability) is an ability to apply accumulated learning skills to the acquisition of new skills (general intelligence, mathematical ability), whereas fluid cognitive ability (specific ability) is a skill that is needed to modify performance requirements (i.e, spatial, reasoning, following directions). Snow (1980) also suggests that when a learning task is novel to the learner, fluid ability is more relevant than previously crystallized learning skills.

Since learning to program is a new experience for most students seeking computer literacy, fluid ability may be more closely related to computer programming success. In addition, numerous studies have correlated programming success and crystallized ability (i.e, mathematical, verbal, academic achievements in mathematic or science) suggesting that mathematical and academic achievement is a strong

predictor of programming success (Peterson & Howe, 1979; Mazlack, 1980; Konvalina, Wileman, & Stephens, 1983; Fisher, & Mandinach, 1985; Webb, 1985; Mayer, Dyck, Vilberg, 1986; McCoy & Burton, 1987).

General ability is often a combination of many specific cognitive abilities. Most computer programming aptitude tests measure general ability, and show a moderate correlation between test scores and programming performance (i.e, IBM Programmers Aptitude Test, Aptitude Assessment Battery-Programming). However, these findings are of little value in developing an instructional treatment to meet the differing needs of individual students. As Webb (1985) points out "...since programming aptitude tests typically assess a combination of abilities, it is difficult to determine whether some abilities are more important than others for predicting performance" (p.184).

Programming and individual differences

Programming is made up of component skills (Shneiderman, 1980; Pea & Kurland, 1983) and each skill may favor a different cognitive ability or cognitive style. Several studies have explored the relationship among cognitive style, cognitive ability and computer programming performance in children and precollege students (Fletcher, 1984; Webb, 1984, 1985; Bradley, 1986; Pommersheim, 1986; McCoy & Burton, 1987). However, only a few studies (Snow, 1980; Mayer, Dyck, Vilberg, 1986; Werth, 1986) have examined the relationship of cognitive style and ability in adults. Snow (1980) correlated diverse individual characteristics with programming outcomes. He found that only two of the many individual characteristics were significantly related to learning BASIC at college level. He concluded that fluid analytical ability was more important than general crystallized ability in predicting a learning outcome.

Mayer, Dyck, & Vilberg also found logical reasoning and spatial ability to be significant factors in learning BASIC. However, they suggested that specific cognitive skills such as direction following, word problem translation, and procedure following were better predictors of learning BASIC. Werth (1986) found that field-independence was significantly related to course grade in an introductory PASCAL course. Since computer programming requires a combination of many skills, examining the relationship between a specific cognitive factor and several programming component skills may impact upon programming instruction. However, only a few studies have examined specific cognitive predictors in regard to programming component skills. Webb (1984, 1985) found that spatial ability was the best predictor for knowledge of LOGO and BASIC commands, and that field-independence was moderately related to programming skills in LOGO, but not in BASIC. These findings provide insight into the cognitive requirement for novice programming, but more research is needed to better understand the relationship between individual differences and programming outcomes. The selected cognitive factors also need further investigation in order

to apply general findings to adult novice programmers, since it is not certain that the cognitive skills of young children and the cognitive skills of adults predict the same outcomes in learning computer programming.

Method

Subjects

Forty-six students who enrolled in a computer literacy course during the fall term of 1987 participated in this study, to include, seventeen graduate students and twenty-nine juniors and seniors in the school of education. Their ages ranged from 21 to 46. Sixteen students reported that they had used word processing, spreadsheet, or other application software, but only six reported that they had any knowledge of programming in BASIC, Pascal, or Logo.

Instrumentation

Cognitive style

Cognitive style was measured by the Group Embedded Figures Test (Oltman, Ruskin, & Witkin, 1971). Subjects were presented with a complex figure and were asked to find a simple figure embedded in the complex one. The tests were 10 minutes long, and consisted of 18 complex pictures. Scores could range from a perfect 18 correct to zero correct. A high score indicates field-independence while a low score indicates field-dependence. Test validity and reliability have been established for the GEFT (Witkin, Oltman, Raskin, & Karp, 1971). Using correlations between parallel forms, a reliability estimate of .82 was determined.

Cognitive ability

The three cognitive ability tests were timed, paper-and-pencil tests, and all were selected from a Kit of Factor-Referenced Cognitive Tests (Ekstrom, French, and Harman, 1976). Because of time constraints, only Part A of each test was used in this study. The manual stated that these tests were valid for grades 9 through adults.

Spatial ability was measured by the Paper Folding Test. In this test, figures were drawn to show each step of paper folding. The last figure contained one or two holes to indicate where the paper had been punched. Subjects were asked to choose one of five figures that showed the correct pattern of hole punches. The test was 3 minutes long and consisted of 10 items.

Logical reasoning was measured by the Letter Sets Test. Five sets of letters were presented with four letters in each set. Four of the sets were alike in some way. Students were asked to find the odd set. The test was a 7-minute timed test and consisted of 15 items. The reliability of the test was .77.

Direction following ability was measured by the Following Directions test. Subjects were asked to determine the correct letter in a matrix of letters by following a complex set of directions (e.g. "Which is the only letter that appears directly above the letter A?"). The test was 7 minutes long and consisted of 10 items.

Programming competence measure

Four programming tests were given throughout the treatment period. Each test included items to assess knowledge of syntax, ability to interpret a program (comprehension), ability to design and generate codes (composition), ability to find mistakes in the existing program (debugging), and the ability to restructure a given program by adding or changing codes (modification). Each test included the five component skills. For example, to test syntax knowledge, students needed to recognize BASIC syntax in a list of correct and incorrect statements. To test comprehension, students predicted outcomes from a short program (6 to 20 lines). To test program composition, students were asked to generate a program after reading one or two paragraphs which described a problem to be solved. To test program debugging, students had to find and correct mistakes in both syntax and logic. To test program modification, a function of an existing program was provided and students were asked to add more programming codes to produce different outcomes.

Each item for programming comprehension, composition, debugging, and modification was scored on a scale of 0 to 10, making partial credit possible. Construct validity was determined by asking three programming instructors to act as judges. They were asked to identify the category of each question based on criteria provided by the researcher. The interjudge agreement was 100% for syntax, comprehension, and composition, and 93% on debugging and modification.

Procedure

Course content consisted of computer application, issues in educational computing, and BASIC programming. Four cognitive pretests, measuring cognitive style and ability, were administered during the 3rd week of the semester. BASIC programming began during the fourth week. Four tests were given to students during the 9-week treatment period. Programming tests were given at intervals for two reasons: First, in order to increase the reliability of subskill tests, several items for each component skill had to be asked. Students would be overwhelmed with

one test. Second, tests served as an instructional activity as well as an evaluation tool.

During the instructional treatment period, students were guided by lecture and a textbook (Computer Literacy: A Hands-On Approach, Luehrmann & Peckham, 1986). Program design, debugging, and modification skills were emphasized as students learned programming concepts. In addition to in-class work, five programming assignments were required for a course grade.

Data Analysis and Results

Out of 46 subjects, 45 completed all cognitive pre-tests and all programming exams. Field-independence was treated as a continuous variable rather than a dichotomous variable. Logical reasoning, spatial ability, and direction following were scored by counting correct answers. A scoring scheme was developed for computer programming items. Syntax answers were judged either true or false, not allowing for partial credit. Scores in other categories were graded on a scale of 0 to 10, so that partial credit could be given for partially correct answers.

To examine the relationships between cognitive variables and computer programming competence, intercorrelations among computer programming subscores, cognitive style, and cognitive ability are presented in Table 1.

Insert Table 1 here

Field-independence was moderately related to all of the computer programming competence scores ($p < .01$). Logical reasoning and direction following were significantly related to most computer programming skills except the syntax score ($p < .001$). Spatial ability was related to programming debugging and modification in this study. Previous findings of a positive relationship between spatial ability and computer programming success was based on a composite score (Snow, 1980; Mayer, Dyck, & Vilberg, 1986). Therefore, it is difficult to compare previous findings to findings in this study. It is interesting that spatial ability is related only to those skills considered complex (debugging, modification). It is reasonable to assume that the ability to transform one image into another may be important in debugging and changing programs.

Although the focus of this study was to determine the relationship between cognitive variables and computer programming competence,

intercorrelations were also examined in order to examine the effect of these factors upon learning computer programming.

Field-independence is related to the other three cognitive ability variables. Spatial ability is related to only field-independence. Reasoning is related to field-independence and direction following. Direction following is related to field-independence and logical reasoning. The strong relationship between direction following and logical reasoning may imply that these two variables were strongly related to all programming skills except the knowledge of syntax.

Relationships between cognitive style/cognitive abilities, and the initial acquisition of computer programming competence were examined. Four conclusions were drawn from data. (1) Logical reasoning and direction following were strongly related to program comprehension, composition, debugging, and modification. (2) Spatial ability is only related to debugging and modification, (3) Cognitive style is moderately related to all of the five programming skills: Syntax, comprehension, composition, debugging, and modification. (4) Cognitive style is related to knowledge of syntax but cognitive ability is not related to knowledge of syntax.

Based on Shneiderman's syntactic and semantic model, programming component skills were subdivided to examine the relationship between the 4 cognitive variables and the 5 computer programming component skills. A strong relationship between logical reasoning and programming coincides with previous findings (Mayer,Dyck,& Vilberg, 1986). This suggests that logical reasoning is similar to processes that are required in programming. According to Wardell (1973), induction is largely a synthesizing process. Program composition, debugging, and modification require the assembling of program statements into a logically sequenced program list. However most novice programmers find it more difficult to write or debug programs than to learn language features (Linn, 1985). Findings of a positive relationship between logical reasoning and all programming skills but syntax seem reasonable, since the knowledge of syntax is acquired by rote learning rather than by applying a synthesizing process.

A positive relationship between direction following and programming is expected because programming involves procedural skills such as sequencing and tracking statements. Direction following is referred to "integrative process" which is an ability to keep in mind simultaneously, or to combine several conditions to produce a correct response (Ekstrom, French, & harman, 1975). Program comprehension, composition, debugging, and modification require close tracking of function of statements as well as their relationship to other statements. Finding of a relationship between direction following and programming is particulary valuable, because following directions is a specific skill that may be trainable than skills of general ability (Mayer, Dyck, Vilberg, 1986). Since

direction following (a procedural skill) seems related to success in learning to program, procedural skills may be developed through instruction. Although procedural skills are employed in many problem solving situations, these skills are not explicitly taught in schools. A nonsignificant relationship with syntax and direction following is also expected because knowledge of syntax is acquired by rote learning.

Field-independence was moderately related to all of the five programming components. The similarity between cognitive processes which field-independent learners employ and cognitive processes that are required for programming anticipates a positive correlation between the two. Indeed, several studies have reported a positive relationship between field-independence and programming (Webb, 1984; Bradley, 1986; Werth, 1986). Webb (1985) and Bradley (1986) found that field-independence was related to LOGO programming achievement scores with small children. Werth (1986) found a significant relationship between field-independence and PASCAL programming success. The current study found a relationship between field-independence and the acquisition of initial programming competence in BASIC with adults. In spite of different languages and age groups, the consistent positive relationship between programming and field-independence suggests that "cognitive restructuring" is important in learning to program. Data from the current study also suggests that field independence seems to underlie other cognitive abilities because of the significant correlation between GEFT scores and logical reasoning, spatial, and direction following.

A positive relationship was found between spatial ability and programming debugging and modification. The current study findings on the relationship between spatial and other component skills (syntax, comprehension, composition) do not support Webb's findings (1984). However, she used LOGO language to examine the relationship between spatial ability and the acquisition of syntax, program interpretation, and program generation. It is possible that adults recognize BASIC syntax more easily than precollege students, because a significant relationship was found between spatial ability and more cognitivly demanding tasks such as program debugging and modification. Also, the conflicting results on program comprehension and composition between Webb's study and the current study may be due to different languages used.

In summary, field-independence, logical reasoning, spatial ability, and direction following were found to be related to computer programming. The importance of logical reasoning and direction following have been found in this study. The lack of a relationship between the knowledge of syntax and cognitive ability should be investigated further in future studies.

References

Ausburn, L. J., & Ausburn, F. B. (1978). Cognitive styles: Some information and implications for instructional design. ECTJ,26(4), 337-354.

Bradley, C. A. (1986). The relationship between students' information-processing styles and Logo programming. Journal of Educational Computing Research, 1(4), 427-432.

EKstrom, R., French, J. W., & Harman. (1976). Kit of reference testsor cognitive factors. Princeton, N.J: Educational Testing Service, 1976.

Fisher, C. & Mandinach, E. (1985). Individual differences and acquisition of computer programming skill. Paper presented at the annual meeting of the American Educational Research Association, Chicago, March 1985.

Fletcher, S. (1984). Cognitive abilities and computer programming. (ERIC Document Reproduction Service No. ED 259 700)

Konvalina, J., Wileman, S. A., & Stephens, L. J. (1983). Math Proficiency: A key to success for computer science students Communications of ACM, 26, 377-382.

Kolb, D. A. (1984). Experiential learning: Experience as the source of learning and development. Englewood Cliffs: Prentice-Hall.

Linn, M. C. (1985). The cognitive consequences of programming instruction in classroom. Educational Researcher, 14(5), 14-29.
Luehrmann, A. (1981). Computer literacy-What should it be?. Mathematics Teacher, 74(9), 682-686.

Luehrmann, A. & Peckham, H. (1986). Computer literacy: A hands-on approach, (rev. ed.). New York: McGraw-Hill.

Mayer, R. E., Dyck, L., Vilberg, W. (1986). Learning to program and learning to think: What's the connection? Communication of the ACM, 20(7), 605-610.

Mazlack, L. (1980). Identifying potential to acquire programming skill. Communications of the ACM, 23, 14-17.

McCoy, L., & Burton, J. (1987). The relationship of computer programming and mathematics in secondary students. Paper presented at AERA annual meeting, Washington, D.C., April 1987.

Messick, M. (1984). The nature of cognitive styles: Problems and promise in educational practice. Educational Psychologist, 19(2), 59-74.

Pea, R. D., & Kurland, D. M. (1983). On the cognitive prerequisites of learning computer programming (Technical Report no. 18). New York: Bank Street College of Education.

Petersen, C. G., & Howe, T. G. (1979). Predicting academic success in introduction to computers. AEDS Journal, 12, 182-191.

Oltman, P. K., Ruskin, E., & Witkin, H. A. (1971). Group embedded figures test. Palo Alto: Consulting Psychologists Press. .

Pommersheim, J. P. (1986). Computer Programming achievement, cognitive styles, and cognitive profiles. AEDS Journal, 19, 51-59.

Shneiderman, B. (1980). Software psychology: Human factors in computer and information systems, Cambridge: Wintrop.

Snow, R. E. (1980). Aptitude processes. In R. E. Snow, P. Federico, and W.E. Montague (Eds.), Aptitude, learning, and instruction. Erlbaum, Hillsdale, N.J.

Soloway, E. (1987). It's 2020: Do you know what your children are learning in programming class? Unpublished manuscript, Yale University, Department of Computer Science, New Haven.

Stevens, D. J. (1983). Cognitive processes and success of students in instructional computer courses. AEDS Journal, 16, 228-233.

Webb, N. M. (1984). Microcomputer learning in small groups: Cognitive requirements and group processes. Journal of Educational Psychology, 76, 1076-1088.

Webb, N. M. (1985). Cognitive requirements of learning computer programming in group and individual settings. AEDS Journal 18, 183-193.

Werth, L. H. (1986). Predicting student performance in a beginning computer science class. ACM, 138-142.

Witkin, H. A., Moore, C.a., Goodenough, D. R., & Cox, Pl W. (1977). Field-dependent and field-independent cognitive styles and their educational implications. Review of Educational Research, 47, 1-64.

Witkin, H. A., Oltman, p. K., Raskin, E., & Karp, S. A. (1971). A manual

for the embedded figure test. Palo Alto: Consulting Psychologists Press

Table 1

Intercorrelations Among Computer Programming Subscores, Cognitive Style, and Cognitive Ability

	Syntax	Compre	Compo	Debug	Modif	Style	Spatial	Reason
Style	.279*	.359**	.385**	.430**	.371**			
Spatial	.027	.236	.250	.288*	.323*	.434**		
Reason	.221	.640***	.528***	.537***	.532**	.468***	.231	
Direction	.244	.600***	.570***	.505***	.462**	.365**	.188	.642***

Note: N=46 * p < .05 ** p < .01 *** p < .001